

## CLAIMS

What is claimed is:

1        1. A method for preventing needless rescanning of objects comprising:  
2                verifying an integrity of a file system by scanning the file system resulting in at least one  
3 known scanned region;  
4                creating a database of the known scanned regions of the verified file system; and  
5                validating an integrity of an object in the file system against the database of known  
6 scanned regions.

1        2. The method of claim 1, wherein verifying the integrity of the file system comprises  
2 scanning the objects in the file system for a presence of viruses.

1        3. The method of claim 1, wherein verifying the integrity of the file system further  
2 comprises monitoring the objects in the file system.

1        4. The method of claim 3, wherein monitoring comprises:  
2                receiving a file system event from a real-time monitoring system, the file system event  
3 indicating that an object in the file system has been accessed; and  
4                flagging the database of known scanned regions to indicate which of the known scanned  
5 regions was occupied by the accessed object.

1        5. The method of claim 4, wherein the database of known scanned regions comprises a copy  
2 of a partition table data structure indicating an identity and a location of a known scanned region  
3 occupied by the object.

1        6. The method of claim 5, wherein the partition table data structure includes an inode that  
2 contains information about the object other than name, and a directory block that contains the  
3 object name and a number of the inode of the object.

1        7. The method of claim 6, wherein the partition table data structure includes a super block  
2 that contains information about the file system as a whole, and a data block that contains a  
3 location in the file system where the object is stored.

1        8. The method of claim 7, wherein validating the integrity of an object comprises  
2 determining that the object does not occupy a flagged known scanned region.

1        9. The method of claim 6 wherein flagging comprises indicating which of the inodes and  
2 directory blocks were occupied by the accessed object.

1        10. The method of claim 9, wherein validating the integrity of an object comprises  
2 determining that the object does not occupy a flagged inode and directory block.

1        11. The method of claim 1 further comprising:

2        rescanning the object when the integrity of the object has not been validated; and

3 bypassing rescanning the object when the integrity of the object has been validated.

1 12. The method of claim 4, wherein the real-time monitoring system is a VxD program.

1 13. The method of claim 4, wherein the real-time monitoring system is a UNIX daemon.

1 14. The method of claim 4, wherein the real-time monitoring system is a network loadable  
2 module.

1 15. An apparatus comprising:  
2 a machine-accessible medium having stored thereon executable instructions to cause a  
3 computer to perform:  
4 verifying an integrity of a file system by scanning the file system resulting in at  
5 least one known scanned region;  
6 creating a database of known scanned regions of the verified file system; and  
7 validating the integrity of an object in the file system against the database of  
8 known scanned regions.

1 16. The apparatus of claim 15, wherein verifying the integrity of the file system comprises  
2 scanning every object in the file system for the presence of viruses.

1 17. The apparatus of claim 15, wherein verifying the integrity of the file system further  
2 comprises monitoring the objects in the file system.

1 18. The apparatus of claim 17, wherein the instructions for monitoring comprises:  
2 receiving a file system event from a real-time monitoring system, the file system event  
3 indicating that an object in the file system has been accessed; and  
4 flagging the database of known scanned regions to indicate which of the known scanned  
5 regions was occupied by the accessed object.

1 19. The apparatus of claim 18, wherein the database of known scanned regions comprises a  
2 copy of a partition table data structure indicating an identity and a location of the known scanned  
3 region occupied by the object.

1 20. The apparatus of claim 19, wherein the partition table data structure includes at least one  
2 of an inode that contains information about the object other than name, and a directory block that  
3 contains the object name and a number of the inode of the object.

1 21. The apparatus of claim 19, wherein the partition table data structure includes a super  
2 block that contains information about the file system as a whole, and a data block that contains a  
3 location in the file system where the object is stored.

1 22. The apparatus of claim 15, wherein validating the integrity of an object comprises  
2 determining that the object does not occupy a flagged region.

1        23. The apparatus of claim 18, wherein flagging comprises indicating which of the inodes  
2        and directory blocks were occupied by the opened or changed object.

1        24. The apparatus of claim 20, wherein the instructions for validating the integrity of an  
2        object comprises determining that the object does not occupy a flagged inode and directory  
3        block.

1        25. The apparatus of claim 15 further comprising instructions to cause a computer to  
2        perform:

3        rescanning the object when the integrity of the object has not been validated; and  
4        bypassing rescanning the object when the integrity of the object has been validated.

1        26. The apparatus of claim 18, wherein the real-time monitoring system is a VxD program.

1        27. The apparatus of claim 18, wherein the real-time monitoring system is a UNIX daemon.

1        28. The apparatus of claim 18, wherein the real-time monitoring system is a network loadable  
2        module.

1        29. An integrity validator comprising:  
2        a verifier to verify the integrity of a file system by scanning the file system resulting in at  
3        least one known scanned region;  
4        a database of the known scanned regions of the verified file system; and

5 a validator to validate the integrity of an object in the file system against the database of  
6 known scanned regions.

1 30. The device of claim 29, wherein the verifier verifies the integrity of the file system by  
2 scanning the objects in the file system for the presence of viruses.

1 31. The device of claim 29, wherein the verifier monitors at least one of the objects in the file  
2 system.

1 32. The device of claim 31, wherein monitors comprises:  
2 receiving a file system event from a real-time monitoring system, the file system event  
3 indicating that an object in the file system has been accessed; and  
4 flagging the database of known scanned regions to indicate which of the known scanned  
5 regions was occupied by the accessed object.

1 33. The device of claim 32, wherein the database of known scanned regions comprises a  
2 copy of a partition table data structure indicating an identity and a location of a known scanned  
3 region occupied by the object.

1 34. The device of claim 33, wherein the partition table data structure includes an inode that  
2 contains information about the object other than name, and a directory block that contains the  
3 object name and a number of the inode of the object.

1 35. The device of claim 33, wherein the partition table data structure includes a super block  
2 that contains information about the file system as a whole, and a data block that contains a  
3 location in the file system where the object is stored.

1 36. The device of claim 29, wherein the validator validates the integrity of an object by  
2 determining that the object does not occupy a flagged known scanned region.

1 37. The device of claim 32, wherein flagging comprises indicating which of the partition  
2 table data structures were occupied by the accessed object.

1 38. The device of claim 34, wherein the validator validates the integrity of an object by  
2 determining that the object does not occupy a flagged partition table data structure.

1 39. The device of claim 29, further comprising:  
2 an AV scanner to rescan the object when the integrity of the object has not been  
3 validated, wherein rescanning is bypassed when the integrity of the object has been validated.

1 40. The device of claim 32, wherein the real-time monitoring system is a VxD program.

1 41. The device of claim 32, wherein the real-time monitoring system is a UNIX daemon.

1 42. The device of claim 32, wherein the real-time monitoring system is a network loadable  
2 module.

1        43. A computer system comprising:  
2            a processor coupled to a system bus;  
3            a memory coupled to the processor through the system bus;  
4            a machine-accessible medium coupled to the processor through the system bus;  
5            an integrity process executed from the machine-accessible medium by the processor,  
6 wherein the integrity process causes the processor to verify the integrity of a file system resulting  
7 in at least one known scanned region, to create a database of known scanned regions of the  
8 verified file system, and to validate the integrity of an object in the file system against the  
9 database of known scanned regions.

1        44. The computer system of claim 43, wherein the system causes the processor to verify the  
2 integrity of the file system by scanning the objects in the file system for the presence of viruses.

1        45. The computer system of claim 44, wherein the system causes the processor to further  
2 verifies the integrity of the file system by monitoring the objects in the file system.

1        46. The computer system of claim 45, wherein the system causes the processor to monitor  
2 the objects in the file system by:  
3            receiving a file system event from a real-time monitoring system, the file system event  
4 indicating that an object in the file system has been accessed; and  
5            flagging the database of known scanned regions to indicate which of the known scanned  
6 regions was occupied by the accessed object.



1        47. The computer system of claim 46, wherein the database of known scanned regions  
2        comprises a copy of a partition table data structure indicating an identity and a location of a  
3        known scanned region occupied by the object.

1        48. The computer system of claim 47, wherein the partition table data structure includes an  
2        inode that contains information about the object other than name, and a directory block that  
3        contains the object name and a number of the inode of the object.

1        49. The computer system of claim 47, wherein the partition table data structure includes a  
2        super block that contains information about the file system as a whole, and a data block that  
3        contains a location in the file system where the object is stored.

1        50. The computer system of claim 43, wherein the system causes the processor to validate the  
2        integrity of an object by determining that the object does not occupy a flagged known scanned  
3        region.

1        51. The computer system of claim 46, wherein the system causes the processor to flag the  
2        database of known regions to indicate which of the inodes and directory blocks were occupied by  
3        the opened or changed object.

1        52. The computer system of claim 48, wherein the system causes the processor to validate the  
2 integrity of an object by determining that the object does not occupy a flagged inode and  
3 directory block.

1        53. The computer system of claim 43, wherein the system further causes the processor to:  
2        rescan the object when the integrity of the object has not been validated; and  
3        bypass the rescan of the object when the integrity of the object has been validated.

1        54. The computer system of claim 46, wherein the real-time monitoring system is a VxD  
2 program.

1        55. The computer system of claim 46, wherein the real-time monitoring system is a UNIX  
2 daemon.

1        56. The computer system of claim 46, wherein the real-time monitoring system is a network  
2 loadable module.